# COMP2200/COMP6200 – Week 9B Scatter plots with stories

Greg Baker

7th October 2025

# Recap and roadmap

# Where we left off

- You can launch Matplotlib, set a figure size, and label the basics.
- You practised a small scatter plot driven by a pandas DataFrame.
- Today we add colour, categories, and legends so the plot tells a story.

# Agenda for this 20 minute sprint

1. Rebuild the iris data set with tidy column names.
2. Plot species with different colours straight from a Series.
3. Customise markers, add a legend, and style titles consistently.

# Loading and tidying data

# Familiar data, Python edition

```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.datasets import load_iris
raw = load_iris(as_frame=True)
iris = raw.frame.rename(
    columns={
        "sepal length (cm)": "sepal_length",
        "sepal width (cm)": "sepal_width",
        "petal length (cm)": "petal_length",
        "petal width (cm)": "petal_width",
        "target": "species",
    }
)
name_dictionary =
    dict(enumerate(raw.target_names))
```

# Familiar data, Python edition

- Same fields you saw in Orange. We just rename them so dot-notation works if you prefer it later.
- The `map` call swaps integers for human-readable species labels.

# Scatter plots with colour

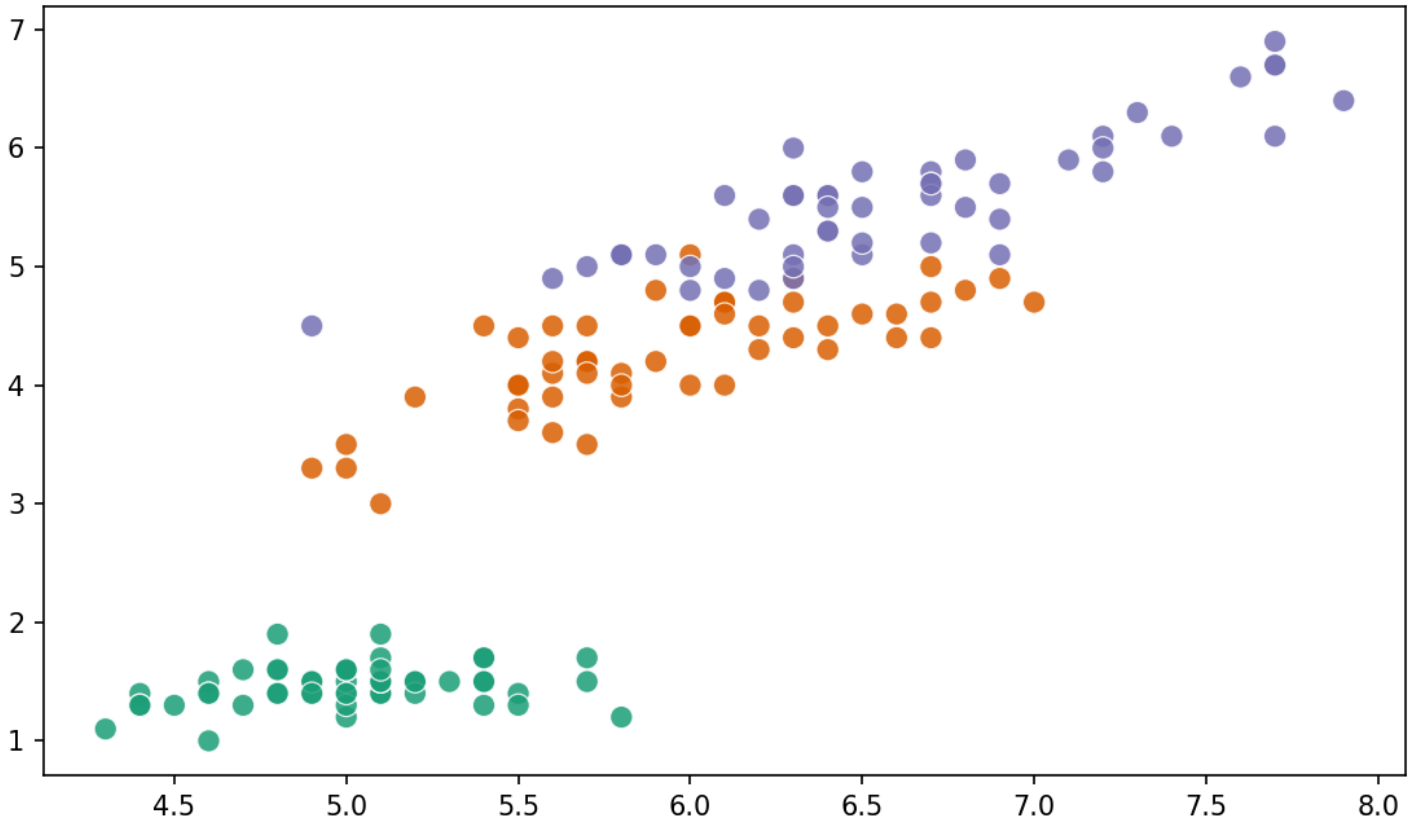
# Mapping categories to colours

```
palette = {
    "setosa": "#1b9e77",
    "versicolor": "#d95f02",
    "virginica": "#7570b3",
}

fig, ax = plt.subplots(figsize=(7.5, 4.5))
colours = iris["species"].map(palette)
scatter = ax.scatter(
    iris["sepal_length"],
    iris["petal_length"],
    c=colours,
    s=70,
    linewidth=0.6,
    edgecolor="white",
```

# Mapping categories to colours

- Passing a Series to `c=` lets pandas broadcast colours row by row.
- A thin white `edgecolor` separates overlapping markers.



# Designing a colour strategy

- Choose hues with clear contrast for the audience (e.g. colourblind-friendly palettes).
- Link colour meanings to the question you are answering (ULO1).
- Explain why the palette emphasises similarities or differences; colour can overstate gaps (ULO4).

**Rule of thumb**

One hue per category, plus a neutral reference, keeps the story honest.

# Using Matplotlib colormaps

- Matplotlib provides built-in colormaps for categorical and continuous data.
- For categorical data, use qualitative colormaps like `tab10`, `Set2`, or `Paired`.
- For continuous data, use sequential (`viridis`, `plasma`) or diverging (`RdBu`, `coolwarm`) colormaps.

## Colorblind-friendly options

Use `viridis`, `plasma`, or ColorBrewer palettes designed for accessibility.

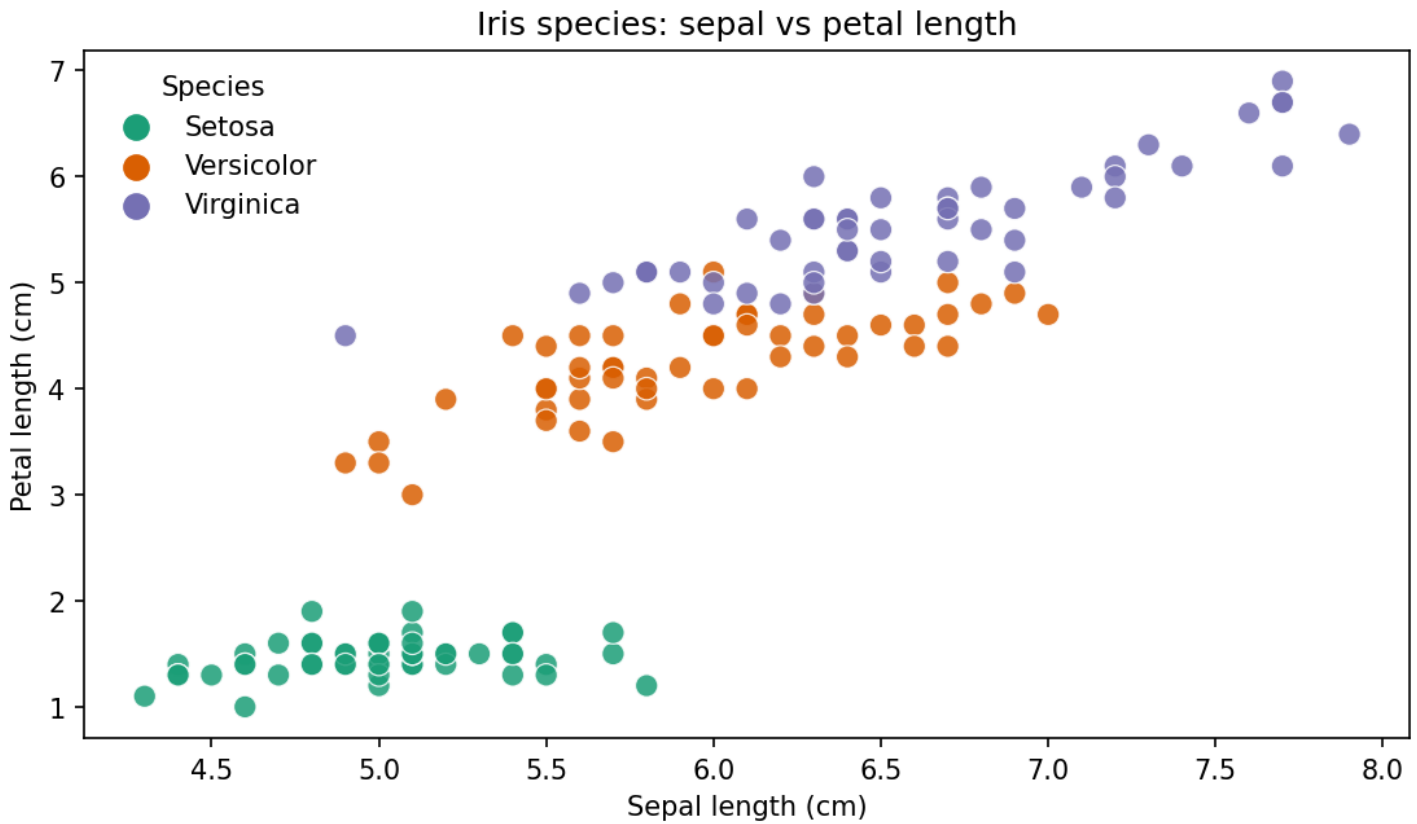
```
import matplotlib.pyplot as plt
# Get colors from a colormap
cmap = plt.cm.Set2
colors = [cmap(i) for i in range(3)]
```

# Legends that explain the palette

```
for species, colour in palette.items():
    ax.scatter([], [], c=colour,
        label=species.title(), s=80)
legend = ax.legend(title="Species",
    frameon=False)
ax.set(
    xlabel="Sepal length (cm)",
    ylabel="Petal length (cm)",
    title="Iris species: sepal vs petal
        length",
)
fig.tight_layout()
fig.savefig(OUTPUT_DIR /
    "9b_colour_legend.png", dpi=150,
    bbox_inches="tight")
```

# Legends that explain the palette

- Empty scatter calls create proxy artists – a neat trick for manual legend entries.
- Hide the legend frame when you want a cleaner slide or report.



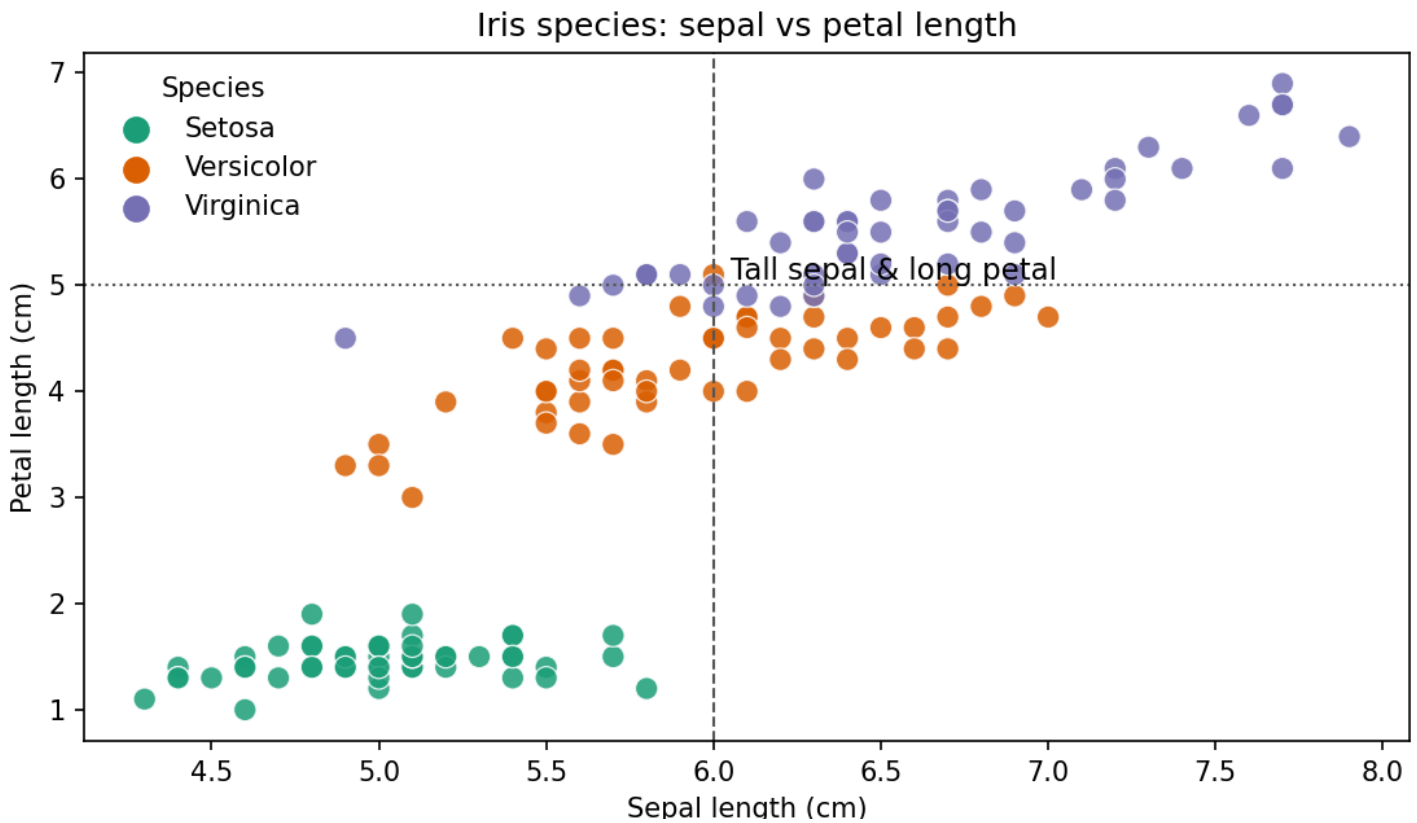
# Adding guidance lines

# Vertical and horizontal cues

```
ax.axvline(x=6.0, color="#555555",
    linestyle="--", linewidth=1)
ax.axhline(y=5.0, color="#555555",
    linestyle=":", linewidth=1)
ax.text(6.05, 5.05, "Tall sepal & long
    petal", fontsize=11)
fig.tight_layout()
fig.savefig(OUTPUT_DIR /
    "9b_guidance_lines.png", dpi=150,
    bbox_inches="tight")
```

# Vertical and horizontal cues

- `axvline` and `axhline` annotate decision boundaries or targets.
- Position text with small nudges so it does not collide with the lines.



# Concluding the scatter story

# Key takeaways

- Colour encodes categories when you feed Matplotlib a Series or list of labels.
- Legends may need manual love; proxy artists give you full control.
- Axes helpers like `axvline` and `axhline` provide quick reference guides for your reader.

# Active design challenge

## Scenario

Your client sells three study-planner apps and wants to compare satisfaction scores across them.

1. Decide how you would colour the products so the preferred option stands out without misleading.
2. Note what you will put in the legend and any guidance lines you would add.
3. Share with a neighbour: how does your palette support the client's decision?

# Next up

- Part C focuses on drawing simple trend lines, comparing subplots, and exporting clean figures.
- Bring any questions from this scatter session – we will recap before pushing ahead.